

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of: )  
Sashikanth Chandrasekaran, et al. )      **Group Art Unit: 2172**  
Serial No.: 09/265,489 )  
Filed: March 9, 1999 )  
For: METHOD AND SYSTEM FOR )  
RELIABLE ACCESS OF MESSAGES BY )  
MULTIPLE CONSUMERS )

---

**DECLARATION OF SASHIKANTH CHANDRASEKARAN AND ASHOK SAXENA  
UNDER 37 C.F.R. § 1.131**

Assistant Commissioner for Patents

Washington, D.C. 20231

Sir:

We, Sashikanth Chandrasekaran and Ashok Saxena, hereby declare as follows:

1. We are co-inventors of the invention described in the above application.
2. At the time we invented the subject invention, we were employed as software and technology developers at Oracle Corporation.
3. As evidenced by the documents attached to this affidavit as Exhibits A-F, prior to January 15, 1999, we had conceived and diligently reduced to practice the subject matter of

the above application. Exhibit A is a portion of a Design Specification dated prior to January 15, 1999 which describes the design specification for implementing database tables and related structures for managing message data to be accessed by multiple recipients. Exhibit B is a copy of a driver script which invokes other SQL test scripts. Exhibit C is a copy of a script that was used to create multiple consumers queues based on the subject invention. Exhibit D are copies of scripts that perform operations such as enqueueing and dequeuing of messages, as well as cleanup of the queue tables. Exhibit E are copies of the outputs generated by the test scripts of Exhibits B-D. Exhibit F are copies of documents showing successful tests of the above scripts.

4. Sections 2.2.4 and Section 3.4 of Exhibit A describes history management processes that are implemented in a software program that was created and reduced to practice prior to January 15, 1999 which embodied the subject matter of the above application. These sections disclose history management of information for multiple consumers, where the information includes one or more information records in order from one or more queues. Section 2.2.4 describes the provision of data from information records to consumers, e.g., by de-queuing a message. This section also describes updating a history table which includes records for the consumers. Section 3.4.1 of Exhibit A describes database fields that are employed to manage history information for the processing of messages, as well as procedures for proving to users and updating such information. Specific information to track for history records are also provided in this section. The fields used to manage the history information includes at least one field that indicates whether a data item has been provided to a consumer, which is updated when a consumer accesses the information (e.g., by updating the "deq\_time" field with the time at which the message was dequeued.). This database structure was implemented in a software program that was created and reduced to practice prior to January 15, 1999 which embodied the subject matter of the above application. Section 3.4.2 of Exhibit A illustrates the algorithm to update and manage index records relating to the messages and message recipients. This process was implemented in the software program that was created and reduced to practice prior to January 15, 1999 which embodied the subject matter of the above application. These sections as well as others within Exhibit A such as section 3.4.1 in the discussion regarding "a dequeue index" and in section 3.4.2 in subsection 1 and at Exhibit E, page 16 (marked as page "1" on the bottom) describe managing information to be accessed by multiple consumers,

where the information includes one or more information records, and the information records to be accessed by the multiple consumers are in a specified order, with each information record including data to be accessed by a consumer. Exhibit E, page 2 (marked as page "2" on the bottom), lines 7 and lines 35-end illustrates the successful creation of the "dequeue index" to allow information records by the multiple consumers to be accessed in a specified order.

4. The attached exhibits disclose providing data of an information record to a consumer. For example, Exhibit E, page 9 (marked as "Page 1" on bottom of page) discloses this, particularly at the "sys.message" statement on or around line 15 of the page and Exhibit E, page 10 (marked as "Page 2" on bottom of page), lines 43-end discloses this, particularly at the "Message" statements.

5. The attached exhibits disclose updating a history table, said history table comprising a history record for said consumer for said information record, said history record comprising a message state field for indicating whether said data of said information record have been provided to said consumer. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table and section 3.4.2 in subsection 3 in the discussion of "array insert..." and in Exhibit C, page 3, at or around lines 45-48, with respect to the statement "If hist(i).transaction\_id IS NOT NULL..."

6. The attached exhibits disclose that said updating comprising setting said message state field in a history record corresponding to said consumer to indicate said consumer accessed said data. For example, For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table and at section 3.4.2 in the discussion regarding "2. Update deq\_time....".

7. The attached exhibits disclose that each said information record further comprises a message identifier value that identifies the data of said information record, and each said history record further comprises a message id field that identifies data in an information record. For example, Exhibit A, section 3.4.1 discloses this in subsection a with respect to the discussion of a history index for a queue table and in Exhibit D, page 3 (marked as "Page 1" on bottom of page)

discloses this, particularly at the “dbms\_aq.enqueue()” statement on or around line 15 of the page relating to “enqmsgid” and at Exhibit D, page 4 (marked as “Page 2” on bottom of page) at lines 26 and 51 with references to the “dbms\_aq.dequeue ()” statements.

8. The attached exhibits disclose that each said history record further comprises a consumer id field that identifies a consumer of said multiple consumers that is to access data in an information record, said data identified by said message id field in said history record, said consumer id field of said history record identifying said history record as corresponding to said consumer. For example, Exhibit A, section 3.4.1 discloses this in subsection e with respect to the discussion of a history index for a queue table and in section 3.4.2 in subsection 3 in the discussion of “array insert...” and in section 3.4.2 in the discussion regarding “2. Update deq\_time....” with respect to “rl” and at Exhibit D, page 4 (marked as “Page 2” on bottom of page) at lines 21 and 47 with references to the “deqopt.consumer\_name := consumer;” statements, Exhibit E, page 13 (marked as “Page 1” on bottom of page) regarding the “dequeue\_options.consumer\_name := subscriber” statement.

9. The attached exhibits disclose that updating comprises setting said message state field in the history record with a message id field that identifies said data that said consumer is provided access to and with a consumer id field that identifies said consumer. For example, Exhibit A, section 3.4.1 discloses this in subsection e with respect to the discussion of a history index for a queue table and in section 3.4.2 in subsection 3 in the discussion of “array insert...” and in section 3.4.2 in the discussion regarding “2. Update deq\_time....” with respect to “rl” and at Exhibit D, page 4 (marked as “Page 2” on bottom of page) at lines 21 and 47 with references to the “deqopt.consumer\_name := consumer;” statements Exhibit E, page 13 (marked as “Page 1” on bottom of page) regarding the “dequeue\_options.consumer\_name := subscriber” statement.

10. The attached exhibits disclose storing data to be accessed by a consumer in an information record. For example, Exhibit D, page 3 (marked as “Page 1” on bottom of page) discloses this, particularly at the “dbms\_aq.enqueue()” statement on or around line 15 of the page relating to “enq\_userdata” and at Exhibit E page 8 (marked as “Page 1” on bottom of page), line 18 regarding the statement “11>....” With respect to the “enq\_userdata” element.

11. The attached exhibits disclose creating a history record for each consumer that is to access said data. For example, Exhibit A, section 3.4.1 discloses this in subsection e with respect to the discussion of a history index for a queue table and in section 3.4.2 in subsection 3 in the discussion of “array insert...”.

12. The attached exhibits disclose setting said message state field in each said history record to indicate said data has not been accessed. For example, Exhibit A, section 3.4.2 discloses this in subsection 3 in the discussion of “array insert...” with respect to the “NULL” elements.

13. The attached exhibits disclose identifying the data of an information record that a consumer is to be provided access to by order data in a read-order table, said order data indicating a relative order that data in said information records is to be accessed by said multiple consumers. For example, Exhibit A, section 3.4.1 discloses this in the discussion regarding “a dequeue index” and in section 3.4.2 in subsection 1 and at Exhibit E, page 16 (marked as page “1” on the bottom) and at Exhibit E, page 2 (marked as page “2” on the bottom), lines 7 and lines 35-end.

14. The attached exhibits disclose reading one or more history records of said history table, said one or more history records comprising a history table read. For example, Exhibit A, section 3.4.1 discloses this with respect to the discussion of a history index for a queue table.

15. The attached exhibits disclose deleting an information record if all the message state fields in all of the history records of said history table read indicate that said data in said information record has been accessed. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

16. The attached exhibits disclose associating a work list table with said history table, said work list table comprising one or more work entries, each said work entry comprising an identification of data in an information record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

17. The attached exhibits disclose adding a work entry to said work list table, said work entry comprising an identification of said data said consumer is provided access to. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

18. The attached exhibits disclose accessing a work entry in said work list table. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

19. The attached exhibits disclose reading one or more history records of said history table, said one or more history records comprising a history table read, said one or more history records comprising said history table read determined by said work entry. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

20. The attached exhibits disclose deleting an information record if all the message state fields in all of the history records of said history table read indicate that said data in said information record has been accessed. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

21. The attached exhibits disclose results from a system and method that implements batching two or more work entries in said work list table. For example, the disclosure of Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities was implemented with batching for the test system that produced the results shown in Exhibits E and F.

22. The attached exhibits disclose reading one or more history records of said history table, said one or more history records determined by said two or more work entries. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

23. The attached exhibits disclose deleting one or more information records. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

24. The attached exhibits disclose an information queue comprising one or more information queue records. For example, this is disclosed in Exhibit A, section 3.4.1 regarding a queue table.

25. The attached exhibits disclose each said information queue record comprising information to be accessed by one or more consumers. For example, this is disclosed in Exhibit A, section 3.4.1 regarding a queue table.

26. The attached exhibits disclose a table separated from said information queue, said table comprising one or more table records, each said table record comprising an identification of said information in an information queue record, each said table record further comprising a consumer identification field comprising an identification of one of said one or more consumers, and a message state field for indicating whether one of the one or more information queue records has been accessed by one of the one or more consumers. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

27. The attached exhibits disclose that each information queue record comprises identification of information of an information queue record. For example, Exhibit A, section 3.4.1 discloses this in subsection a with respect to the discussion of a history index for a queue table and in Exhibit D, page 3 (marked as "Page 1" on bottom of page) discloses this, particularly at the "dbms\_aq.enqueue() statement on or around line 15 of the page relating to "enqmsgid".

28. The attached exhibits disclose a read-order table, said read-order table comprising order data indicating the order that information in said information queue is to be delivered to a consumer. For example, this is disclosed in Exhibit A, section 3.4.2 regarding the discussion of "1. insert one key...".

29. The attached exhibits disclose that a read-order table comprises one or more records, each said record of said read-order table comprising an identification field that identifies

information in an information queue record, each said record of said read-order table further comprising an enqueue time field that comprises said order data. For example, this is disclosed in Exhibit A, section 3.4.2, regarding the discussion of “1. insert one key...” and regarding “1. Delete its index entry from qt\_i.”

30. The attached exhibits disclose a work list table, said work list table comprising one or more work list entries, each said work list entry comprising an identification of information in an information queue record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

31. The attached exhibits disclose that a work list entry is a record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

32. The attached exhibits disclose that a work list table comprises one or more work records and each said work list entry is a field in a work record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

33. The attached exhibits disclose a message queue comprising one or more message queue records, each said one or more message queue records comprising a message and a message identification. For example, this is disclosed in Exhibit A, section 3.4.1 regarding a queue table.

34. The attached exhibits disclose a history table separated from said message queue comprising one or more history records, each of said one or more history records comprising a message identification, a consumer identification and a message state identification, each said message state identification indicating whether one of the one or more message queue records has been accessed. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

35. The attached exhibits disclose a work list table separated from said message queue and said history table comprising one or more work list entries, each said work list entry comprising a message identification. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

36. The attached exhibits disclose a method for multiple consumers to access information in a non first-in first-out, prescribed order, said information comprising one or more pieces of information, a first piece of information stored in a first location. For example, this is disclosed in Exhibit C, page 2, line 1-10, particularly with respect to the "sort\_list" element(s) of the statement(s).

37. The attached exhibits disclose providing access to first piece of information to a first consumer of said multiple consumers; indicating in a second location in a history table that said first consumer has accessed said first piece of information, said history table having a first message state field for indicating whether said first consumer has accessed said first piece of information; providing access to said first piece of information to a second consumer of said multiple consumers; and indicating in a third location in said history table that said second consumer has accessed said first piece of information, said history table having a second message state field for indicating whether said second consumer has accessed said first piece of information. For example, this is disclosed in Exhibit E, page 11, lines 25-end and at page 17, lines 1-24.

38. The attached exhibits disclose that a first location comprises an information entry in a queue of information. For example, this is disclosed in Exhibit E, pages 1-3.

39. The attached exhibits disclose a queue of information comprises one or more information entries, and each said information entry comprises a piece of information to be accessed by one or more of said multiple consumers, each said information entry further comprising an identification of said piece of information in said information entry. For example, this is disclosed in Exhibit E, pages 1-3.

40. The attached exhibits disclose deleting an entry comprising a first piece of information that a first consumer and a second consumer is provided access to from said queue of information after said first consumer and said second consumer have accessed said first piece of information. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

41. The attached exhibits disclose for each of one or more consumers, a table comprises a separate table record for each piece of information to be accessed by said consumer. For example, this is disclosed in Exhibit A, section 3.4.1 and section 3.4.2.

42. The attached exhibits disclose the history table comprises an identification of a first piece of information and an identification of a first consumer. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

43. The attached exhibits disclose a history entry in said history table, said history entry comprising an identification of a first piece of information and an identification of a second consumer. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

44. The attached exhibits disclose indicating in a location an order in which one or more pieces of information is to be accessed by multiple consumers. For example, this is disclosed in Exhibit C, page 2, line 1-10, particularly with respect to the "sort\_list" element(s) of the statement(s).

45. The subject invention was reduced to practice and tested to verify that it works for its intended purpose prior to January 15, 1999. This is shown by Exhibits B-F which include copies of documents which evidence that the subject invention was tested and found to work for its intended purpose. As noted above, Exhibit B is a copy of a driver script which invokes the other SQL test scripts. Exhibit C is a copy of a script that was used to create multiple consumers queues based on the subject invention. Exhibit D are copies of scripts that perform operations such

as enqueueing and dequeuing of messages, as well as cleanup of the queue tables. Exhibit E are copies of the outputs generated by the test scripts of Exhibits B-D. Exhibit F show copies of documents showing successful tests of the above scripts. Regression tests were performed to verify that the operations performed by the test scripts generate the specified output. The documents in Exhibit F are copies of files that display the results of running the tests and demonstrate that the tests ran correctly with the expected results.

13. We further declare that all statements made herein of my own knowledge are true and all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: June 21 2007



Sashikanth Chandrasekaran

Patent  
237/116  
OI7011472001

Date: June 22, 2007

Ashok Saxena

Ashok Saxena

**EXHIBIT A**

---

# Design Specification for AQ Propagation , RDBMS, 8.1

Project ID: aq\_propagation

Version: [REDACTED]

Status: Approved

Author: Sashi Chandrasekaran, Ashok Saxena

[REDACTED]

[REDACTED]

Version	Reviewers	Changes
[REDACTED]	arsaxena	Creation
[REDACTED]	[REDACTED] [REDACTED] [REDACTED] [REDACTED] [REDACTED]	[REDACTED]

#### **2.2.4 History Management**

History management for multi-consumer queues leaves a lot to be desired. There are two fundamental problems to history management: storage and reference counting.

- **Storage**  
The history information is stored as a varray object collection. The current interface to varray collections retrieves and unpickles the entire collection. AQ uses C interfaces to navigate through the collection and update the history element for the appropriate consumer. The updated collection is written entirely into the database for history tracking.
  - **Reference Counting**

In Oracle 8.0.4 a reference count is maintained as a separate column with each message. Dequeuers decrement the reference count and the last dequeuer (that finds the reference count to be zero) deletes the message from the queue table. Needless to say when several consumers are trying to dequeue the same messages a convoy will quickly form behind the reference count hotspot. We intend to eliminate the hot spot by de-coupling the reference counting from the dequeuers by entrusting the queue monitor with the task of performing the garbage collection (i.e delete messages that have been dequeued by all consumers). It already performs the task of deleting messages that have expired.

We propose to keep the history information in a separate IOT. This will reduce the contention among the multiple consumers to update the history and eliminate the need to lock the queue table entry for the message. The time manager index will be enhanced and the time manager's responsibilities will be increased to update the state of the message to "PROCESSED". If the queue has a non-infinite retention time, the time-manager will not update the state of the message to processed. Instead, the time-manager will only remove the message when the retention time is complete. This is not a problem because the state of the message can be deduced from the history entries in the history IOT. When a message is dequeued by a consumer, its entry in the message table is updated to the new removal time (current time + retention). When the time manager encounters this entry it will check the history and if all recipients have processed the message it will remove the message from the queue table and also the history rows. Since the history IOT is indexed on message id as the leading primary key the time manager can efficiently determine the status of a message.



## 3.4 History Management

### 3.4.1 Data Structures

When a queue table is created, three additional IOTs are created to store the message meta data. The message data and message properties are kept in the queue table. The queue table in 8.1 will be modified for changes in message properties, but they are not relevant to the history management. The three additional IOTs are:

- A dequeue index to maintain the sort-order of messages for each recipient. This index will have the same structure as in Oracle 8.0.
- A history index that maintains the history of processing of every message. The columns in this IOT are as follows:

- a. msgid - unique identifier of the message
- b. rowid - location of the message in the queue table.
- c. address - address of the recipient.

It is the source queue name (without the schema name appended to it) if the consumer will dequeue messages directly from the source queue. The address supported by AQ propagation will be of the form [schema.]queue[@database\_link]. Messages are propagated to the destination queue specified by the address. AQ does not require global names to be set to TRUE, however it is recommended. The database link name is resolved in the context of the owner of the source queue.

- d. protocol - protocol field of the recipient structure.

This field qualifies the address. It is the session-level protocol (e.g. dblink/TIB) used to propagate messages to the destination queue. It is 0 if the address is a database link address or if the consumer dequeues the message from the local queue.

- e. consumer\_name - name of the agent (recipient) that dequeued the message.

- f. txn\_id - transaction id of the dequeuing transaction.

- g. deq\_time - time of dequeue.

- h. deq\_user - database schema id of dequeuer.

- j. propagated\_mgid - message id of the enqueued message in the destination queue.

This is NULL if the address is NULL.

- k. retry\_count - # times message was dequeued in remove mode (and aborted).

Columns a, c, d and e form the primary key. We may choose to include the other columns also as part of the primary key to simplify access to these columns (Key columns are easier to extract than non-key columns and also do not have the complexity of an overflow segment). Key-compression will not be used since we do not expect the prefix (msgid) to be repeated often.

- A time-manager index that maintains the list of time-management activities. The time-manager index has four columns:

- a. time - absolute time at which time-manager has to perform an operation.

- b. msgid - message id of message that needs to be acted upon.
  - c. action - a description of the action that needs to be performed. The possible values are:
    - 1. MAKE\_READY - make message available for dequeue to consumers after the delay time has passed.
    - 2. EXPIRE - move message to exception queue if message has not yet been processed.
    - 3. REMOVE - remove message after the retention time has passed.
  - d. transaction\_id. This is the transaction\_id of the transaction that inserts the time-management entry. This is needed to generate a unique key, since two consumers can dequeue the same message and post the time-manager to perform an action at the same time. This is set if the action is REMOVE.
- Columns a, b and d form the primary key. This IOT is similar to the time-manager index for Oracle 8.0 queue tables. The differences are:
- a. The IOT stores the msgid of the message rather than its rowid.
  - b. There is an action column to help the time-manager determine what time-management activity needs to be performed on the message. In theory, this column is superfluous because the time-manager can deduce what action needs to be performed based on the history information in the historytable. Oracle8.0'stime-managerindexdeducedwhataction needs to be performed based on the state of the message in the queue table.
  - c. There could be multiple rows for the same message in the index. In fact, there could be up to one row for each agent that dequeues the message from the queue table. This is because each agent that dequeues messages independently notifies the time-manager without knowledge of the state of the message with respect to other recipients.

### 3.4.2 Design Description

We illustrate the use of these index structures using a simple example. Let us assume that a queue table, say qt, has been created. Call the dequeue sort order index qt\_i, the history index as qt\_h and the time-manager index as qt\_t. Let us say a message is enqueued in queue q with the following properties: messageid = m, delay = d, expiration = e, retention time = r, recipients = {r1, r2@boston} where r1 is a local consumer and boston is a remote database. The acknowledgment mode for this message is assumed to be ACK\_DEQUEUED (a propagator and dequeuer perform similar actions if the acknowledgment mode is ACK\_PROPAGATED or NO\_ACK).

When the message is enqueued at rowid = rid, the index structures are updated as follows:

1. insert one key into qt\_i for the propagator. This step is identical to Oracle 8.0. This step is necessary so that the propagator can dequeue the message without waiting for the delay time.
2. If  $disnon-NULL$  insert key into qt\_t with value [d, m, MAKE\_READY, txnid] else if e is non-NULL insert key into qt\_t with value [e, m, EXPIRE, txnid]
3. array insert two keys into qt\_h with values [m, rid, r2@boston, 0, r2, NULL, NULL, NULL, NULL, 0] and [m, rid, q, 0, r1, NULL, NULL, NULL, NULL, 0]. This step will substitute generating the history collection in an 8.0 queue table.

When the delay time has passed the time-manager performs the following actions.

1. for each entry in qt\_h where msgid = m and address = q and txn\_id = txnid insert key into qt\_i to enable consumer to dequeue message.
2. update the qt\_t key to [e, m, EXPIRE, cur\_txnid] if e is non-NULL.

Agent r1 performs the following steps after dequeuing message m.

1. Delete its index entry from qt\_i.

---

**3 Design Description****Design Specification for <Feature>, <Product>, <Product Version>**

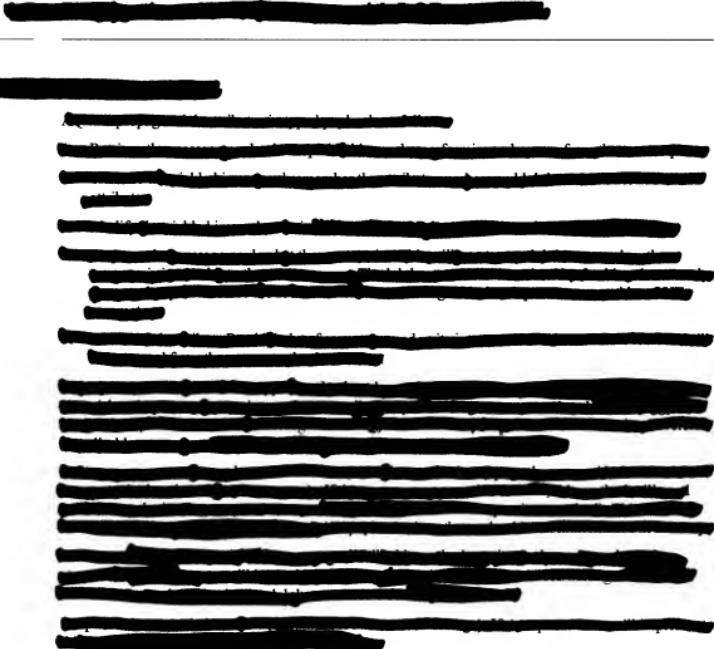
---

2. Update `deq_time`, `deq_user`, `txn_id` columns in `qt_h` for row with `consumer_name = r1`.
3. If `retention_time` is not `NULL`, insert key `[r, m, REMOVE, cur_txnid]` into `qt_t` else if queue has no retention, insert key `[gettimeofday(), m, REMOVE, cur_txnid]`.

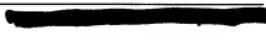
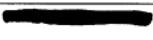
The propagator updates the propagated `_msgid` column in `qt_h` and deletes the index entry from `qt_i` as soon as `m` is successfully propagated to `boston`. The `deq_time` column in `qt_h` and `time_manager` index `qt_t` are updated only on receipt of acknowledgment from `boston` that `r2` has processed the message.

The time-manager marks the message as expired at time `e`, if either `r1`'s or `r2`'s `deq_time` columns in `qt_h` is `NULL`. Likewise, it removes the message `m` at time `r` only if `r1` and `r2` `deq_time` columns are `non-NULL`. In all cases the time-manager removes the index entry from the `time_manager_index` when it processes the entry, regardless of whether processing it resulted in any state change or not. When a message expires, the history keys in `qt_h` are copied over to a different queue table if the exception queue resides in a different queue table. The history keys are deleted along with the message itself when the application uses `dequeue-by-message-id` to remove the message from the exception queue.

---



---



**EXHIBIT B**

```
# [REDACTED]
# [REDACTED]
# tkaqnsbd.tsc
# [REDACTED]
# [REDACTED]
# NAME
#   tkaqnsbd.tsc - Advanced Queueing New-style Short Basic Dequeue test
# DESCRIPTION
#   short regression test of 8.1 style multiconsumer queues
# NOTES
#   tests creation, navigation, enqueue, dequeue
# [REDACTED]
# [REDACTED]
# [REDACTED]
# [REDACTED]
# RUNS_STANDALONE Yes
# TEST_TYPE Short
# USES SQL
# MAX_USERS 1
#
# open_cursors, shared_pool_size settings are temporary workarounds
# rdbmsini 07_DICTIONARY_ACCESSIBILITY=true compatible=8.1.0 db_block_buffers=400
open_cursors=500 shared_pool_size=8000000
get tkaqmin.sql
sql tkaqmin > tkaqnsb1
compare tkaqnsb1 tkaqmin mask
# Enqueue some messages and navigate thru the queues
get tkaqnav.sql
sql tkaqnav > tkaqnsb2
compare tkaqnsb2 tkaqnav mask
# Dequeue some messages when there is more than one message for an application
# First create the messages in the queue.
get tkaqmbdm.sql
sql tkaqmbdm > tkaqnsb3
compare tkaqnsb3 tkaqmbdm mask
get tkaqnsd1.sql
sql tkaqnsd1 > tkaqnsb4
compare tkaqnsb4 mask
# Cleanup
get tkaqmcln.sql
sql tkaqmcln > tkaqnsb5
compare tkaqnsb5 mask
shutdown
```

**EXHIBIT C**

```
Rem
Rem
Rem
Rem
Rem tkaqnmin.sql
Rem
Rem
Rem
Rem NAME
Rem tkaqnmin.sql - <one-line expansion of the name>
Rem
Rem DESCRIPTION
Rem <short description of component this file declares/defines>
Rem
Rem NOTES
Rem <other useful comments, qualifications, etc.>
Rem
Rem
Rem
Rem
Rem
Rem
Rem
```

```
set echo on
connect sys/knl_test7 as sysdba
```

```
Rem
Rem Create a queue user and administrator
```

```
grant connect,resource,aq_administrator_role to tkaqadmnn identified by tkaqadmnn;
grant connect,resource,aq_user_role to tkaquser identified by tkaquser;

grant execute on dbms_aq to tkaquser;
execute dbms_aqadm.grant_system_privilege('ENQUEUE_ANY','tkaquser',FALSE);
execute dbms_aqadm.grant_system_privilege('DEQUEUE_ANY','tkaquser',FALSE);
grant execute on dbms_aqadm to tkaquser;
execute dbms_aqadm.grant_system_privilege('MANAGE_ANY','tkaquser',TRUE);
grant execute on dbms_aq to tkaqadmnn;
execute dbms_aqadm.grant_system_privilege('ENQUEUE_ANY','tkaqadmnn',FALSE);
execute dbms_aqadm.grant_system_privilege('DEQUEUE_ANY','tkaqadmnn',FALSE);
execute dbms_aqadm.grant_type_access('tkaqadmnn');
```

```
Rem
Rem Create a type
```

```
create type message as object(id NUMBER, data VARCHAR2(30));
```

```
grant execute on message to tkaqadmnn;
grant execute on message to tkaquser;
```

```
Rem TODO: the remaining SQL scripts must be executed as tkaqadmnn
Rem connect tkaqadmnn/tkaqadmnn
```

```
# Create a queue with default sort ordering
```

```
execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtdef',
queue_payload_type => 'message', multiple_consumers => true, comment => 'Creating
queue table with default sort ordering', compatible => '8.1.3');
```

```

# Create a queue with priority and enq_time as the sort order
execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtpeqt',
queue_payload_type => 'message', sort_list => 'priority,enq_time', multiple_consumers => true, comment => 'Creating queue with priority and enq_time sort order', compatible => '8.1.3');

# Create a queue with priority as the sort order
execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtprt',
queue_payload_type => 'message', sort_list => 'priority', multiple_consumers => true, comment => 'Creating queue with priority sort order', compatible => '8.1.3');

Rem CHECK IF ALL'S OK
select schema, name, flags objno from system.aq$_queue_tables;
select orderbypos, colno, name, sort_order, table_objno from system.aq$_queue_table_sort order by table_objno, orderbypos;

Rem Ensure that the tables and the indices have been created
select count(*) from tkaqqtdef;
select count(*) from aq$_tkaqqtdef_i;

select count(*) from tkaqqtpeqt;
select count(*) from aq$_tkaqqtpeqt_i;

select count(*) from tkaqqtprt;
select count(*) from aq$_tkaqqtprt_i;

Rem create two queues in each queue table
Rem create two queues in tkaqqtdef
execute dbms_aqadm.create_queue(queue_name => 'q1def', queue_table => 'sys.tkaqqtdef', max_retries => 2, comment => 'queue 1 in tkaqqtdef');
execute dbms_aqadm.create_queue(queue_name => 'q2def', queue_table => 'sys.tkaqqtdef', max_retries => 2, comment => 'queue 2 in tkaqqtdef');

Rem create two queues in tkaqqtpeqt
execute dbms_aqadm.create_queue(queue_name => 'q1peqt', queue_table => 'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 1 in tkaqqtpeqt');
execute dbms_aqadm.create_queue(queue_name => 'q2peqt', queue_table => 'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 2 in tkaqqtpeqt');

Rem create two queues in tkaqqtprt
execute dbms_aqadm.create_queue(queue_name => 'q1pri', queue_table => 'sys.tkaqqtprt', max_retries => 2, comment => 'queue 1 in tkaqqtprt');
execute dbms_aqadm.create_queue(queue_name => 'q2pri', queue_table => 'sys.tkaqqtprt', max_retries => 2, comment => 'queue 2 in tkaqqtprt');

Rem create an exception queue in the tkaqqtdef table
execute dbms_aqadm.create_queue(queue_name => 'exceptionq', queue_table => 'sys.tkaqqtdef', queue_type => DBMS_AQADM.EXCEPTION_QUEUE, comment => 'exception q in tkaqqtdef');

```

Rem Create procedure to check the list of subscribers for each queue

```
CREATE OR REPLACE PROCEDURE TKAQ_SUBSCRIBERS(qname VARCHAR2) AS
```

```
subs      dbms_aqadm.aq$_subscriber_list_t;
nsub      BINARY_INTEGER;
i        BINARY_INTEGER;

begin

  subs := dbms_aqadm.queue_subscribers(qname);

  dbms_output.put_line(qname);
  dbms_output.put_line('-----');

  nsub := subs.COUNT;
  FOR i IN 0..nsub-1 LOOP
    IF subs(i) IS NOT NULL THEN
      dbms_output.put_line('--> ' || subs(i).name);
    END IF;
  END LOOP;
```

```
end TKAQ_SUBSCRIBERS;
```

```
/
```

```
CREATE OR REPLACE PROCEDURE TKAQ_HISTORY(enqmsgid RAW) AS
```

```
hist      sys.aq$_history;
nsub      BINARY_INTEGER;
i        BINARY_INTEGER;
```

```
begin
```

```
  select history into hist
  from  tkaggtddef
  where  msgid = enqmsgid;
```

```
  dbms_output.put_line('-----');
  IF hist IS NOT NULL THEN
    nsub := hist.COUNT;
    dbms_output.put_line(nsub);
    FOR i IN 1..nsub LOOP
      IF hist(i) IS NOT NULL THEN
        dbms_output.put_line('APP --> ' || hist(i).consumer);
        IF hist(i).transaction_id IS NOT NULL THEN
          dbms_output.put_line('TXN --> ' || hist(i).transaction_id);
          dbms_output.put_line('DEQ_USER --> ' || hist(i).deq_user);
        ELSE
          dbms_output.put_line('TXN --> ');
        END IF;
      END IF;
    END LOOP;
  END IF;
  dbms_output.put_line('-----');
```

```
end TKAQ_HISTORY;
```

```
/
```

Rem check subscribers for queues created

```
set serveroutput on
```

[REDACTED]

```
execute tkaq_subscribers('Q1DEF');
execute tkaq_subscribers('Q2DEF');
execute tkaq_subscribers('Q1PEQT');
execute tkaq_subscribers('Q2PEQT');
execute tkaq_subscribers('Q1PRI');
execute tkaq_subscribers('Q2PRI');

Rem add some default subscribers for each queue.

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app1_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',app1_qldef);
end;
/

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app2_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',app1_qldef);
end;
/

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app3_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',app1_qldef);
end;
/

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app4_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',app1_qldef);
end;
/

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app5_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',app1_qldef);
end;
/

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app6_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',app1_qldef);
end;
/

declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_.agent('app1_q2def', NULL, NULL);
```

```
dbms_aqadm.add_subscriber('sys.q2def',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app2_q2def', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2def',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app1_q1peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1peqt',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app2_q1peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1peqt',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app1_q2peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2peqt',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app2_q2peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2peqt',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app1_q1pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1pri',app1_q1def);
end;
/
declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_.agent('app2_q1pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1pri',app1_q1def);
end;
/
```

```
declare
appl_q1def sys.aq$_agent;
begin
appl_q1def := sys.aq$_agent('app1_q2pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2pri',appl_q1def);
end;
/
declare
appl_q1def sys.aq$_agent;
begin
appl_q1def := sys.aq$_agent('app2_q2pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2pri',appl_q1def);
end;
/
Rem check subscribers for queues created
execute tkaq_subscribers('Q1DEF');
execute tkaq_subscribers('Q2DEF');
execute tkaq_subscribers('Q1PEQT');
execute tkaq_subscribers('Q2PEQT');
execute tkaq_subscribers('Q1PRI');
execute tkaq_subscribers('Q2PRI');

Rem start the queues
execute dbms_aqadm.start_queue(queue_name => 'sys.q1def');
execute dbms_aqadm.start_queue(queue_name => 'sys.q2def');
execute dbms_aqadm.start_queue(queue_name => 'sys.q1peqt');
execute dbms_aqadm.start_queue(queue_name => 'sys.q2peqt');
execute dbms_aqadm.start_queue(queue_name => 'sys.q1pri');
execute dbms_aqadm.start_queue(queue_name => 'sys.q2pri');
```

## **EXHIBIT D**





Rem  
Rem [REDACTED]  
Rem  
Rem tkaqmnav.sql  
Rem  
Rem [REDACTED]  
Rem  
Rem NAME  
Rem tkaqmnav.sql - <one-line expansion of the name>  
Rem  
Rem DESCRIPTION  
Rem <short description of component this file declares/defines>  
Rem  
Rem NOTES  
Rem <other useful comments, qualifications, etc.>  
Rem  
Rem [REDACTED]  
Rem  
set echo on  
connect tkaquser/tkaquser  
set serveroutput on  
  
create or replace procedure tkaq\_navenq(priority in number) as  
enq\_userdata sys.message;  
enqmsgid raw(16);  
enqopt dbms\_aq.enqueue\_options\_t;  
msgprop dbms\_aq.message\_properties\_t;  
  
begin  
  
enq\_userdata := sys.message(priority, 'HELLO, WORLD!');  
msgprop.priority := priority;  
dbms\_aq.enqueue('sys.q2pri', enqopt, msgprop, enq\_userdata, enqmsgid);  
end;  
/  
  
execute tkaq\_navenq(1);  
execute tkaq\_navenq(2);  
commit;  
  
execute tkaq\_navenq(3);  
execute tkaq\_navenq(4);  
commit;  
  
execute tkaq\_navenq(5);  
execute tkaq\_navenq(6);  
commit;  
  
execute tkaq\_navenq(7);  
execute tkaq\_navenq(8);  
commit;  
  
execute tkaq\_navenq(9);  
execute tkaq\_navenq(10);  
commit;  
  
execute tkaq\_navenq(11);  
execute tkaq\_navenq(12);

commit;

execute tkaq\_navenq(13);  
execute tkaq\_navenq(14);  
commit;

execute tkaq\_navenq(15);  
execute tkaq\_navenq(16);  
commit;

create or replace procedure tkaq\_navdeq(consumer IN VARCHAR2) as  
deq\_userdata sys.message;  
deqmsgid raw(16);  
deqopt dbms\_aq.dequeue\_options\_t;  
msgprop dbms\_aq.message\_properties\_t;

begin

deqopt.wait := DBMS\_AQ.NO\_WAIT;  
deqopt.consumer\_name := consumer;  
deqopt.navigation := DBMS\_AQ.FIRST\_MESSAGE;  
deqopt.dequeue\_mode := DBMS\_AQ.BROWSE;

FOR i in 1..9 loop  
dbms\_aq.dequeue('sys.q2pri', deqopt, msgprop, deq\_userdata, deqmsgid);  
dbms\_output.put\_line('Message: ' || deq\_userdata.id  
|| ':' || deq\_userdata.data);  
commit;  
deqopt.navigation := DBMS\_AQ.NEXT\_MESSAGE;  
END LOOP;

end;  
/  
execute tkaq\_navdeq('app1\_q2pri');  
execute tkaq\_navdeq('app2\_q2pri');

create or replace procedure tkaq\_navdeq(consumer IN VARCHAR2) as  
deq\_userdata sys.message;  
deqmsgid raw(16);  
deqopt dbms\_aq.dequeue\_options\_t;  
msgprop dbms\_aq.message\_properties\_t;

begin

deqopt.wait := DBMS\_AQ.NO\_WAIT;  
deqopt.consumer\_name := consumer;  
deqopt.navigation := DBMS\_AQ.FIRST\_MESSAGE;

FOR i in 1..19 loop  
dbms\_aq.dequeue('sys.q2pri', deqopt, msgprop, deq\_userdata, deqmsgid);  
dbms\_output.put\_line('Message: ' || deq\_userdata.id  
|| ':' || deq\_userdata.data);  
commit;  
deqopt.navigation := DBMS\_AQ.NEXT\_MESSAGE;  
END LOOP;

end;  
/  
execute tkaq\_navdeq('app1\_q2pri');  
execute tkaq\_navdeq('app2\_q2pri');

```
set echo on
connect sys/knl_test7 as sysdba
set serveroutput on
```

```
CREATE OR REPLACE PROCEDURE TKAQ_MBASICENQ(id IN NUMBER, text IN VARCHAR2) AS
msgprop      dbms_aq.message_properties_t;
enqopt       dbms_aq.enqueue_options_t;
enq_mmsgid   raw(16);
enq_userdata message;
```

begin

```
    enqueue_name => 'sys.q2def',
    enqueue_options => enqopt,
    message_properties => msgprop,
    payload => enq_userdata,
    msqid => enq_msqid);
```

end;

```
CREATE OR REPLACE PROCEDURE TKAQ_MBASICDEQ(subscriber IN VARCHAR2) AS
  dequeue_options dbms_aq.dequeue_options_t;
  message_properties dbms_aq.message_properties_t;
  deque_userdata sys.message;
  deque_msaid raw(16);
```

```
[REDACTED]  
dequeue_options.consumer_name := subscriber;  
dequeue_options.navigation := DBMS_AQ.FIRST_MESSAGE;  
dequeue_options.wait := 1;  
dbms_aq.dequeue(queue_name=> 'sys.q2def',  
                 dequeue_options=>dequeue_options,  
                 message_properties=>message_properties,  
                 payload=>deq_userdata,  
                 msgid=>deq_msgid);  
commit;  
dbms_output.put_line('MESG-> ' || deq_userdata.id || ' ' || deq_userdata.data);  
end;  
/  
Rem enqueue twelve messages each subscriber should get two.  
execute tkaq_mbasicenq(1, 'First Message');  
execute tkaq_mbasicenq(2, 'Second Message');  
execute tkaq_mbasicenq(3, 'Third Message');  
execute tkaq_mbasicenq(4, 'Fourth Message');  
execute tkaq_mbasicenq(5, 'Fifth Message');  
execute tkaq_mbasicenq(6, 'Sixth Message');  
execute tkaq_mbasicenq(7, 'Seventh Message');  
execute tkaq_mbasicenq(8, 'Eight Message');  
execute tkaq_mbasicenq(9, 'Ninth Message');  
execute tkaq_mbasicenq(10, 'Tenth Message');  
execute tkaq_mbasicenq(11, 'Eleventh Message');  
execute tkaq_mbasicenq(12, 'Twelveth Message');  
execute tkaq_mbasicenq(13, 'Thirteenth Message');  
execute tkaq_mbasicenq(14, 'Fourteenth Message');  
execute tkaq_mbasicenq(15, 'Fifteenth Message');  
execute tkaq_mbasicenq(16, 'Sixteenth Message');  
execute tkaq_mbasicenq(17, 'Seventeenth Message');  
execute tkaq_mbasicenq(18, 'Eighteenth Message');  
execute tkaq_mbasicenq(19, 'Nineteenth Message');  
execute tkaq_mbasicenq(20, 'Twentyth Message');  
execute tkaq_mbasicenq(21, 'Twentyfirst Message');  
execute tkaq_mbasicenq(22, 'Twenty2nd Message');  
execute tkaq_mbasicenq(23, 'Twenty3rd Message');
```

[REDACTED]

```
execute tkaq_mbasicenq(24, 'Twenty4th Message');
commit;
```

**EXHIBIT E**

Echo [REDACTED] ON

```
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR>
SVRMGR>
SVRMGR> Rem
SVRMGR> Rem   Create a queue user and administrator
SVRMGR>
SVRMGR> grant connect,resource,aq_administrator_role to tkaqadmn identified by
tkaqadmn;
Statement processed.
SVRMGR>
SVRMGR> grant connect,resource,aq_user_role to tkaquser identified by tkaquser;
Statement processed.
SVRMGR>
SVRMGR> grant execute on dbms_aq to tkaquser;
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('ENQUEUE_ANY','tkaquser',FALSE);
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('DEQUEUE_ANY','tkaquser',FALSE);
Statement processed.
SVRMGR> grant execute on dbms_aqadm to tkaquser;
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('MANAGE_ANY','tkaquser',TRUE);
Statement processed.
SVRMGR> grant execute on dbms_aq to tkaqadmn;
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('ENQUEUE_ANY','tkaqadmn',FALSE);
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('DEQUEUE_ANY','tkaqadmn',FALSE);
Statement processed.
SVRMGR> execute dbms_aqadm.grant_type_access('tkaqadmn');
Statement processed.
SVRMGR>
SVRMGR> Rem
SVRMGR> Rem   Create a type
SVRMGR>
SVRMGR> create type message as object(id NUMBER, data VARCHAR2(30));
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> grant execute on message to tkaqadmn;
Statement processed.
SVRMGR> grant execute on message to tkaquser;
Statement processed.
SVRMGR>
SVRMGR> Rem TODO: the remaining SQL scripts must be executed as tkaqadmn
SVRMGR> Rem connect tkaqadmn/tkaqadmn
SVRMGR>
SVRMGR> # Create a queue with default sort ordering
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtdef',
queue_payload_type => 'message', multiple_consumers => true, comment => 'Creating
queue table with default sort ordering', compatible => '8.1.3');
Statement processed.
SVRMGR>
SVRMGR> # Create a queue with priority and enq_time as the sort order
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtpeqt',
queue_payload_type => 'message', sort_list => 'priority,enq_time', multiple_consumers
=> true, comment => 'Creating queue with priority and enq_time sort order',
compatible => '8.1.3');
```

Statement processed.

SVRMGR>

SVRMGR> # Create a queue with priority as the sort order

SVRMGR>

SVRMGR> execute dbms\_aqadm.create\_queue\_table(queue\_table => 'sys.tkaqqtpri', queue\_payload\_type => 'message', sort\_list => 'priority', multiple\_consumers => true, comment => 'Creating queue with priority sort order', compatible => '8.1.3');

Statement processed.

SVRMGR>

SVRMGR> Rem CHECK IF ALL'S OK

SVRMGR>

SVRMGR> select schema, name, flags objno from system.aq\$\_queue\_tables;

SCHEMA	NAME	OBJNO
SYSTEM	DEF\$_AQCALL	0
SYSTEM	DEF\$_AQERROR	0
SYS	TKAQQTDEF	9
SYS	TKAQQTPEQT	9
SYS	TKAQQTPRI	9

5 rows selected.

SVRMGR> select orderbypos, colno, name, sort\_order, table\_objno from system.aq\$\_queue\_table\_sort order by table\_objno, orderbypos;

select orderbypos, colno, name, sort\_order, table\_objno from system.aq\$\_queue\_table\_sort order by table\_objno, orderbypos

\*

ORA-00942: table or view does not exist

SVRMGR>

SVRMGR> Rem Ensure that the tables and the indices have been created

SVRMGR>

SVRMGR> select count(\*) from tkaqqtdef;

COUNT(\*)

-----

0

1 row selected.

SVRMGR> select count(\*) from aq\$\_tkaqqtdef\_i;

COUNT(\*)

-----

0

1 row selected.

SVRMGR>

SVRMGR> select count(\*) from tkaqqtpeqt;

COUNT(\*)

-----

0

1 row selected.

SVRMGR> select count(\*) from aq\$\_tkaqqtpeqt\_i;

COUNT(\*)

-----

0

1 row selected.

SVRMGR>

SVRMGR> select count(\*) from tkaqqtpri;

COUNT(\*)

-----

0

1 row selected.

SVRMGR> select count(\*) from aq\$\_tkaqqtpri\_i;

COUNT(\*)

-----

0

1 row selected.

SVRMGR>

SVRMGR> Rem create two queues in each queue table

```

SVRMGR>
SVRMGR> Rem create two queues in tkaqqtdef
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q1def', queue_table =>
'sys.tkaqqtdef', max_retries => 2, comment => 'queue 1 in tkaqqtdef');
Statement processed.
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q2def', queue_table =>
'sys.tkaqqtdef', max_retries => 2, comment => 'queue 2 in tkaqqtdef');
Statement processed.
SVRMGR>
SVRMGR> Rem create two queues in tkaqqtpeqt
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q1peqt', queue_table =>
'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 1 in tkaqqtpeqt');
Statement processed.
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q2peqt', queue_table =>
'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 2 in tkaqqtpeqt');
Statement processed.
SVRMGR>
SVRMGR> Rem create two queues in tkaqqtprt
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'qlpri', queue_table =>
'sys.tkaqqtprt', max_retries => 2, comment => 'queue 1 in tkaqqtprt');
Statement processed.
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q2pri', queue_table =>
'sys.tkaqqtprt', max_retries => 2, comment => 'queue 2 in tkaqqtprt');
Statement processed.
SVRMGR>
SVRMGR> Rem create an exception queue in the tkaqqtdef table
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'exceptionq', queue_table =>
'sys.tkaqqtdef', queue_type => DBMS_AQADM.EXCEPTION_QUEUE, comment => 'exception q
in tkaqqtdef');
Statement processed.
SVRMGR>
SVRMGR> Rem Create procedure to check the list of subscribers for each queue
SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_SUBSCRIBERS(qname VARCHAR2) AS
 2>
 3>   subs    dbms_aqadm.aq$_subscriber_list_t;
 4>   nsubs  BINARY_INTEGER;
 5>   i      BINARY_INTEGER;
 6>
 7> begin
 8>
 9>   subs := dbms_aqadm.queue_subscribers(qname);
10>
11>   dbms_output.put_line(qname);
12>   dbms_output.put_line('-----');
13>
14>   nsubs := subs.COUNT;
15>   FOR i IN 0..nsubs-1 LOOP
16>     IF subs(i) IS NOT NULL THEN
17>       dbms_output.put_line('--> ' || subs(i).name);
18>     END IF;
19>   END LOOP;
20>
21> end TKAQ_SUBSCRIBERS;
22> /
Statement processed.

```

```
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_HISTORY(enqmsgid RAW) AS
2>
3> hist          sys.aq$_history;
4> nsubs          BINARY_INTEGER;
5> i              BINARY_INTEGER;
6>
7> begin
8>
9>   select history into hist
10>  from  tkaqtddef
11>  where  msgid = enqmsgid;
12>
13>
14>   dbms_output.put_line('-----');
15>   IF hist IS NOT NULL THEN
16>     nsubs := hist.COUNT;
17>     dbms_output.put_line(nsubs);
18>     FOR i IN 1..nsubs LOOP
19>       IF hist(i) IS NOT NULL THEN
20>         dbms_output.put_line('APP --> ' || hist(i).consumer);
21>         IF hist(i).transaction_id IS NOT NULL THEN
22>           dbms_output.put_line('TXN --> ' || hist(i).transaction_id);
23>           dbms_output.put_line('DEQ_USER --> ' || hist(i).deq_user);
24>         ELSE
25>           dbms_output.put_line('TXN --> ');
26>         END IF;
27>       END IF;
28>     END LOOP;
29>   END IF;
30>   dbms_output.put_line('-----');
31>
32> end TKAQ_HISTORY;
33> /
```

MGR-00072: Warning: PROCEDURE TKAQ\_HISTORY created with compilation errors.

```
SVRMGR>
SVRMGR>
SVRMGR> Rem check subscribers for queues created
SVRMGR>
SVRMGR> set serveroutput on
Server Output          ON
SVRMGR>
SVRMGR> execute tkaq_subscribers('Q1DEF');
Statement processed.
Q1DEF
```

```
-----  
SVRMGR> execute tkaq_subscribers('Q2DEF');
Statement processed.
Q2DEF
```

```
-----  
SVRMGR> execute tkaq_subscribers('Q1PEQT');
Statement processed.
Q1PEQT
```

```
-----  
SVRMGR> execute tkaq_subscribers('Q2PEQT');
Statement processed.
Q2PEQT
```

```
-----  
SVRMGR> execute tkaq_subscribers('Q1PRI');
Statement processed.
Q1PRI
```

```
-----  
SVRMGR> execute tkaq_subscribers('Q2PRI');
```

Statement processed.  
Q2PRI

```
SVRMGR>
SVRMGR> Rem add some default subscribers for each queue.
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
 3> begin
 4> appl_qldef := sys.aq$_agent('app1_qldef', NULL, NULL);
 5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
 6> end;
 7> /
Statement processed.  
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
 3> begin
 4> appl_qldef := sys.aq$_agent('app2_qldef', NULL, NULL);
 5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
 6> end;
 7> /
Statement processed.  
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
 3> begin
 4> appl_qldef := sys.aq$_agent('app3_qldef', NULL, NULL);
 5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
 6> end;
 7> /
Statement processed.  
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
 3> begin
 4> appl_qldef := sys.aq$_agent('app4_qldef', NULL, NULL);
 5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
 6> end;
 7> /
Statement processed.  
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
 3> begin
 4> appl_qldef := sys.aq$_agent('app5_qldef', NULL, NULL);
 5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
 6> end;
 7> /
Statement processed.  
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
 3> begin
 4> appl_qldef := sys.aq$_agent('app6_qldef', NULL, NULL);
 5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
 6> end;
 7> /
Statement processed.  
SVRMGR>
SVRMGR>
SVRMGR> declare
 2> appl_qldef sys.aq$_agent;
```

```
3> begin
4> appl_qldef := sys.aq$_agent('app1_q2def', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2def',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q2def', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2def',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_q1peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q1peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q1peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q1peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_q2peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q2peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_qlpri', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qlpri',appl_qldef);
6> end;
7> /
Statement processed.
```

```

SVRMGR>
SVRMGR> declare
2> appl_q1def sys.aq$_agent;
3> begin
4>   appl_q1def := sys.aq$_agent('app2_q1pri', NULL, NULL);
5>   dbms_aqadm.add_subscriber('sys.q1pri',appl_q1def);
6> end;
7> /
Statement processed.

SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_q1def sys.aq$_agent;
3> begin
4>   appl_q1def := sys.aq$_agent('app1_q2pri', NULL, NULL);
5>   dbms_aqadm.add_subscriber('sys.q2pri',appl_q1def);
6> end;
7> /
Statement processed.

SVRMGR>
SVRMGR> declare
2> appl_q1def sys.aq$_agent;
3> begin
4>   appl_q1def := sys.aq$_agent('app2_q2pri', NULL, NULL);
5>   dbms_aqadm.add_subscriber('sys.q2pri',appl_q1def);
6> end;
7> /
Statement processed.

SVRMGR>
SVRMGR>
SVRMGR> Rem check subscribers for queues created
SVRMGR>
SVRMGR> execute tkaq_subscribers('Q1DEF');
Statement processed.

Q1DEF
-----
--> APP1_Q1DEF
--> APP2_Q1DEF
--> APP3_Q1DEF
--> APP4_Q1DEF
--> APP5_Q1DEF
--> APP6_Q1DEF
SVRMGR> execute tkaq_subscribers('Q2DEF');
Statement processed.

Q2DEF
-----
--> APP1_Q2DEF
--> APP2_Q2DEF
SVRMGR> execute tkaq_subscribers('Q1PEQT');
Statement processed.

Q1PEQT
-----
--> APP1_Q1PEQT
--> APP2_Q1PEQT
SVRMGR> execute tkaq_subscribers('Q2PEQT');
Statement processed.

Q2PEQT
-----
--> APP1_Q2PEQT
--> APP2_Q2PEQT
SVRMGR> execute tkaq_subscribers('Q1PRI');
Statement processed.

Q1PRI

```

-----  
--> APP1\_Q1PRI  
--> APP2\_Q1PRI  
SVRMGR> execute tkaq\_subscribers('Q2PRI');  
Statement processed.  
Q2PRI  
-----  
--> APP1\_Q2PRI  
--> APP2\_Q2PRI  
SVRMGR>  
SVRMGR>  
SVRMGR> Rem start the queues  
SVRMGR> execute dbms\_aqadm.start\_queue(queue\_name => 'sys.q1def');  
Statement processed.  
SVRMGR> execute dbms\_aqadm.start\_queue(queue\_name => 'sys.q2def');  
Statement processed.  
SVRMGR> execute dbms\_aqadm.start\_queue(queue\_name => 'sys.q1peqt');  
Statement processed.  
SVRMGR> execute dbms\_aqadm.start\_queue(queue\_name => 'sys.q2peqt');  
Statement processed.  
SVRMGR> execute dbms\_aqadm.start\_queue(queue\_name => 'sys.q1pri');  
Statement processed.  
SVRMGR> execute dbms\_aqadm.start\_queue(queue\_name => 'sys.q2pri');  
Statement processed.  
SVRMGR>

Echo [REDACTED] ON  
SVRMGR> connect tkaquser/tkaquser  
Connected.  
SVRMGR> set serveroutput on  
Server Output ON  
SVRMGR>  
SVRMGR> create or replace procedure tkaq\_navenq(priority in number) as  
2> enq\_userdata sys.message;  
3> enqmsgid raw(16);  
4> enqopt dbms\_aq.enqueue\_options\_t;  
5> msgprop dbms\_aq.message\_properties\_t;  
6>  
7> begin  
8>  
9> enq\_userdata := sys.message(priority, 'HELLO, WORLD!');  
10> msgprop.priority := priority;  
11> dbms\_aq.enqueue('sys.q2pri', enqopt, msgprop, enq\_userdata, enqmsgid);  
12> end;  
13> /  
Statement processed.  
SVRMGR>  
SVRMGR> execute tkaq\_navenq(1);  
Statement processed.  
SVRMGR> execute tkaq\_navenq(2);  
Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>  
SVRMGR> execute tkaq\_navenq(3);  
Statement processed.  
SVRMGR> execute tkaq\_navenq(4);  
Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>  
SVRMGR> execute tkaq\_navenq(5);  
Statement processed.  
SVRMGR> execute tkaq\_navenq(6);  
Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>  
SVRMGR> execute tkaq\_navenq(7);  
Statement processed.  
SVRMGR> execute tkaq\_navenq(8);  
Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>  
SVRMGR> execute tkaq\_navenq(9);  
Statement processed.  
SVRMGR> execute tkaq\_navenq(10);  
Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>  
SVRMGR> execute tkaq\_navenq(11);  
Statement processed.  
SVRMGR> execute tkaq\_navenq(12);  
Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>

[REDACTED]

```
SVRMGR> execute tkaq_navenq(15);
Statement processed;
SVRMGR> execute tkaq_navenq(14);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(15);
Statement processed.
SVRMGR> execute tkaq_navenq(16);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> create or replace procedure tkaq_navdeq(consumer IN VARCHAR2) as
2>     deq_userdata sys.message;
3>     deq_msgid    raw(16);
4>     deqopt       dbms_aq.dequeue_options_t;
5>     msgprop      dbms_aq.message_properties_t;
6>
7> begin
8>
9>     deqopt.wait := DBMS_AQ.NO_WAIT;
10>    deqopt.consumer_name := consumer;
11>    deqopt.navigation := DBMS_AQ.FIRST_MESSAGE;
12>    deqopt.dequeue_mode := DBMS_AQ.BROWSE;
13>
14>    FOR i in 1..9 loop
15>        dbms_aq.dequeue('sys.q2pri', deqopt, msgprop, deq_userdata, deq_msgid);
16>        dbms_output.put_line('Message: ' || deq_userdata.id
17>                             || ':' || deq_userdata.data);
18>
19>        commit;
20>        deqopt.navigation := DBMS_AQ.NEXT_MESSAGE;
21>    END LOOP;
22>
23> end;
23> /
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navdeq('appl_q2pri');
Statement processed.
Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!
SVRMGR> execute tkaq_navdeq('app2_q2pri');
Statement processed.
Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!
SVRMGR>
SVRMGR> create or replace procedure tkaq_navdeq(consumer IN VARCHAR2) as
```

```

2>    deq_userdata sys.message;
3>    deq_mgid    raw(16);
4>    deqopt     dbms_aq.dequeue_options_t;
5>    msgprop    dbms_aq.message_properties_t;
6>
7> begin
8>
9>    deqopt.wait := DBMS_AQ.NO_WAIT;
10>   deqopt.consumer_name := consumer;
11>   deqopt.navigation := DBMS_AQ.FIRST_MESSAGE;
12>
13>   FOR i in 1..19 loop
14>      dbms_aq.dequeue('sys.q2pri', deqopt, msgprop, deq_userdata, deq_mgid);
15>      dbms_output.put_line('Message: ' || deq_userdata.id
16>                            || ':' || deq_userdata.data);
17>      commit;
18>      deqopt.navigation := DBMS_AQ.NEXT_MESSAGE;
19>   END LOOP;
20>
21> end;
22> /
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navdeq('app1_q2pri');
ORA-25228: timeout in dequeue from SYS.Q2PRI while waiting for a message
ORA-06512: at "SYS.DBMS_AQ", line .*
ORA-06512: at "TKAQUSET.TKAQ_NAVDEQ", line .*
ORA-06512: at Line .*
Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!
Message: 10:HELLO, WORLD!
Message: 11:HELLO, WORLD!
Message: 12:HELLO, WORLD!
Message: 13:HELLO, WORLD!
Message: 14:HELLO, WORLD!
Message: 15:HELLO, WORLD!
Message: 16:HELLO, WORLD!
SVRMGR> execute tkaq_navdeq('app2_q2pri');
ORA-25228: timeout in dequeue from SYS.Q2PRI while waiting for a message
ORA-06512: at "SYS.DBMS_AQ", line .*
ORA-06512: at "TKAQUSET.TKAQ_NAVDEQ", line .*
ORA-06512: at Line .*
Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!
Message: 10:HELLO, WORLD!
Message: 11:HELLO, WORLD!
Message: 12:HELLO, WORLD!
Message: 13:HELLO, WORLD!
Message: 14:HELLO, WORLD!

```

[REDACTED]  
Message: 15:HELLO, WORLD!  
Message: 16:HELLO, WORLD!  
SVRMGR>

```
Echo                                     ON
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR> set serveroutput on
Server Output          ON
SVRMGR>
SVRMGR>
SVRMGR>
SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_MBASICENQ(id IN NUMBER, text IN VARCHAR2)
AS
2> msgprop      dbms_aq.message_properties_t;
3> enqopt       dbms_aq.enqueue_options_t;
4> enq_mgid    raw(16);
5> enq_userdata message;
6>
7> begin
8>
9>   enq_userdata := message(id, text);
10>  dbms_aq.enqueue(
11>    queue_name => 'sys.q2def',
12>    enqueue_options => enqopt,
13>    message_properties => msgprop,
14>    payload => enq_userdata,
15>    msgid => enq_mgid);
16>
17> end;
18> /
Statement processed.
SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_MBASICDEQ(subscriber    IN VARCHAR2) AS
2>
3>  dequeue_options  dbms_aq.dequeue_options_t;
4>  message_properties dbms_aq.message_properties_t;
5>  deq_userdata      sys.message;
6>  deq_mgid         raw(16);
7> begin
8>  dequeue_options.consumer_name := subscriber;
9>  dequeue_options.navigation := DBMS_AQ.FIRST_MESSAGE;
10> dequeue_options.wait := 1;
11> dbms_aq.dequeue(queue_name=> 'sys.q2def',
12>                  dequeue_options=>dequeue_options,
13>                  message_properties=>message_properties,
14>                  payload=>deq_userdata,
15>                  msgid=>deq_mgid);
16>  commit;
17>
18> dbms_output.put_line('MESG-> ' || deq_userdata.id || ' ' || deq_userdata.data);
19>
20> end;
21> /
Statement processed.
SVRMGR>
SVRMGR> Rem enqueue twelve messages each subscriber should get two.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(1, 'First Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(2, 'Second Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(3, 'Third Message');
```

Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(4, 'Fourth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(5, 'Fifth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(6, 'Sixth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(7, 'Seventh Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(8, 'Eight Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(9, 'Ninth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(10, 'Tenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(11, 'Eleventh Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(12, 'Twelfth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(13, 'Thirteenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(14, 'Fourteenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(15, 'Fifteenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(16, 'Sixteenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(17, 'Seventeenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(18, 'Eighteenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(19, 'Nineteenth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(20, 'Twentyth Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(21, 'TwentyFirst Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(22, 'Twenty2nd Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(23, 'Twenty3rd Message');  
Statement processed.  
SVRMGR> execute tkaq\_mbasicenq(24, 'Twenty4th Message');

Statement processed.  
SVRMGR> commit;  
Statement processed.  
SVRMGR>



Echo [REDACTED] ON  
SVRMGR> connect sys/knl\_test7 as sysdba  
Connected.  
SVRMGR> set serveroutput on  
Server Output ON  
SVRMGR>  
SVRMGR> Rem now do dequeues  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed  
MESG-> 1 First Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 1 First Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 2 Second Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 2 Second Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 3 Third Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 3 Third Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 4 Fourth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 4 Fourth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 5 Fifth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 5 Fifth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 6 Sixth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 6 Sixth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 7 Seventh Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 7 Seventh Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 8 Eight Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 8 Eight Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 9 Ninth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 9 Ninth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.

MESG-> 10 Tenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 10 Tenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 11 Eleventh Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 11 Eleventh Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 12 Twelfth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 12 Twelfth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 13 Thirteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 13 Thirteenth Message  
SVRMGR> commit;  
Statement processed.  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 14 Fourteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 15 Fifteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 16 Sixteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 17 Seventeenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 18 Eighteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 19 Nineteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 20 Twentyth Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 21 Twentyfirst Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 22 Twenty2nd Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 23 Twenty3rd Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
Statement processed.  
MESG-> 24 Twenty4th Message  
SVRMGR> execute tkaq\_mbasicdeq('app1\_q2def');  
ORA-25228: timeout in dequeue from SYS.Q2DEF while waiting for a message  
ORA-06512: at "SYS.DBMS\_AQ", line.\*  
ORA-06512: at "SYS.TKAQ\_MBASICDEQ", line .\*  
ORA-06512: at line.\*  
SVRMGR> commit;

Statement processed.  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 14 Fourteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 15 Fifteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 16 Sixteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 17 Seventeenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 18 Eighteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 19 Nineteenth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 20 Twentieth Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 21 Twentyfirst Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 22 Twenty2nd Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 23 Twenty3rd Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
Statement processed.  
MESG-> 24 Twenty4th Message  
SVRMGR> execute tkaq\_mbasicdeq('app2\_q2def');  
ORA-25228: timeout in dequeue from SYS.Q2DEF while waiting for a message  
ORA-06512: at "SYS.DBMS\_AQ", line.\*  
ORA-06512: at "SYS.TKAQ\_MBASICDEQ", line .\*  
ORA-06512: at line.\*  
SVRMGR> commit;  
Statement processed.

Echo [REDACTED] ON

SVRMGR> connect sys/knl\_test7 as sysdba  
Connected.

SVRMGR> Rem cleanup the queue tables

SVRMGR> execute dbms\_aqadm.drop\_queue\_table('sys.tkaqqtdef', TRUE);  
Statement processed.

SVRMGR> execute dbms\_aqadm.drop\_queue\_table('sys.tkaqqtpdq', TRUE);  
Statement processed.

SVRMGR> execute dbms\_aqadm.drop\_queue\_table('sys.tkaqqtpri', TRUE);  
Statement processed.

SVRMGR>

SVRMGR> Rem connect as sys and drop the types

SVRMGR> connect sys/knl\_test7 as sysdba  
Connected.

SVRMGR>

SVRMGR> drop type message;  
Statement processed.

SVRMGR>

SVRMGR> Rem disable aq logins

SVRMGR> drop user tkaquser cascade;  
Statement processed.

SVRMGR> drop user tkaqadmnn cascade;  
Statement processed.

SVRMGR>

SVRMGR>

execute tkaq\_mbasicdeq('app1\_q2def');

execute tkaq\_mbasicdeq('app1\_q2def');

commit;

execute tkaq\_mbasicdeq('app2\_q2def');

commit;

**EXHIBIT F**

Short regress has 0 dif - run with use\_ism=false.

tk0irdd1.suc	tkagrawt.suc	tkoolgyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrseb.suc
tk0irmts.suc	tkdrplsl.suc	tkogbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkagqbd2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtrres.suc
tkagbqe1.suc	tkhodbk.suc	tkpasr8l.suc	tkprdep3.suc	tkqrwhes.suc
tkagbqe2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkagqdb1.suc	tkhoucmr.suc	tkpbdl.suc	tkprdep5.suc	tkrcrash.suc
tkagqdb2.suc	tkigfsrt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkagqdba.suc	tkirens.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkagdqrc.suc	tkirinds.suc	tkppro.suc	tkprmisc.suc	tkrmliti.suc
tkagqbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsqlp.suc	tkrsubs.suc
tkaqmbe1.suc	tkoodcta.suc	tkpgcaf.suc	tkprttime.suc	tktrclius.suc
tkaqmnav.suc	tkoodctn.suc	tkpdups.suc	tkprtimmm.suc	tktrmnps.suc
tkaqnsb1.suc	tkoodtcl.suc	tkpqgrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpjjois.suc	tkqrccats.suc	tktrrtabs.suc
tkaqnsb3.suc	tkoodtcm.suc	tkpqmain.suc	tkqrdat.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpqord.suc	tkqrrexp.suc	txxabrhc.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqgrgs.suc	tkqrgrps.suc	tkzrlso0.suc
tkaqoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	
tkaqrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	

Long regress will not be started as we need machines for other tests.

Transaction: ntang\_bug-733938

Transaction: ykunitom\_bug-704908\_1

kpodp.c@@/main/3  
 [REDACTED] Bug 704908  
 kdhl.c@@/main/115  
 [REDACTED] d: bug 704908  
 kclbl.h@@/main/26  
 [REDACTED]  
 kia.c@@/main/3  
 [REDACTED] Bug 704908  
 klc.c@@/main/102  
 [REDACTED]  
 kcli.c@@/main/16  
 [REDACTED]  
 opiul.c@@/main/35

Transaction: pong\_bug-713950\_1

ksdx0.h@@/main/4

ksdx.c@@/main/22

**Transaction: weiwang\_bug-718209**

kkdl.c@@/main/378

kkfi.c@@/main/18 Merge From /vobs/rdbms/src/server/dict/dictlkup/kkdl.c@@/main/377

kkfi.c@@/main/17 Merge From /vobs/rdbms/src/server/optim/cbo/kkfi.c@@/main/17

kda.c@@/main/117

kau.c@@/main/97

qerlt.c@@/main/41

qkdrv.c@@/main/224

qkdrv.c@@/main/223 Merge From /vobs/rdbms/src/server/sqlexec/rwsalloc/qkdrv.c@@/main/223

klcbs.h@@/main/47

klcbs.h@@/main/47 check for disabled flag

klc.c@@/main/101

**Transaction: ato\_make\_user\_queue\_table\_views**

catqueue.sql@@/main/34

catqueue.sql@@/main/34 create all\_queue\_tables views

Makefile@@/main/9

Makefile@@/main/9 add prvtaqin.sql

prvtaqin.sql@@/main/3

prvtaqin.sql@@/main/3 add java interface

**Transaction: smuralid\_bug-718348**

ttccap.c@@/main/4

ttccapSnd: don't send if null caps

koka.c@@/main/38

koka.c@@/main/38 Merge From /vobs/rdbms/src/server/objsupp/objdata/koka.c@@/main/37

kpo.h@@/main/21

kpo.h@@/main/21 make kpoRTcaps an SGA variable

opidrv.c@@/main/81

opidrv, opiscb: set hsmrtrcap to null if SGA's not initialized

opiino.c@@/main/35

opiino: set hsmrtrcap to kpoRTcaps after mapping in SGA

opirip.c@@/main/30

opirip.c@@/main/30 Merge From /vobs/rdbms/src/server/progint/opi/opirip.c@@/main/29

opitsk.c@@/main/67

opitsk.c@@/main/67 Merge From /vobs/rdbms/src/server/progint/opi/opitsk.c@@/main/66

**Transaction: rshaikh\_fix\_downgrade**

c0801030.sql@@/main/10

c0801030.sql@@/main/10 drop type rowset

d0800050.sql@@/main/34

[REDACTED] dont drop ustats  
d0801030.sql@@/main/7  
[REDACTED] : fix

Transaction: najain\_658136-1

uacdef.h@@/main/20  
[REDACTED] remove UACFNLMX  
kxs.c@@/main/151  
[REDACTED] remove UACFNLBV  
kks.c@@/main/316  
[REDACTED] remove UACFNLBV  
opix.c@@/main/82  
[REDACTED] remove UACFNLBV

Transaction: thchang\_templob\_rdwr

kokl.c@@/main/64  
[REDACTED] Merge From /vobs/rdbms/src/server/objsupp/objdata/kokl.c@@/main/63  
kok13.c@@/main/21  
[REDACTED] templob

Transaction: ramkrish\_bug-685852\_2

opitsk.c@@/main/68  
[REDACTED] opitsk: bug 685852 - BFILE check

Transaction: lkaplan\_row\_cln\_err

AllA.java@@/main/9  
[REDACTED]  
RefGrp.java@@/main/5  
RepAPI.java@@/main/9  
[REDACTED]

Transaction: sichandr\_bug-729312

ktk.c@@/main/108  
[REDACTED] : fix error message for DATABASE triggers

Transaction: awitkows\_bug-732881

vop.c@@/main/135  
[REDACTED] complex view & correlated var

Transaction: najain\_bug-730182

kql.c@@/main/240  
[REDACTED] : fix 730182

Transaction: heneman\_lrg27341

kmc.c@@/main/100  
[REDACTED] LRG 27341: correct order of interruptable test

**Transaction: ntang\_bug-737564**

k2g.c@@/main/52

[REDACTED] b

Out of Transaction changes:

.depprod@@/main/solaris/227

[REDACTED] : update plsql, precomp, sqlplus, ordts, ldap

.ldapmsg@@/main/264

[REDACTED] : new log

.manifest@@/main/solaris/271

[REDACTED] : new label

mesg@@/main/5

[REDACTED] Add new qsmXX.msg translations for 8.1.4

Added file element "qsmd.msg".

Added file element "qsmf.msg".

Added file element "qsmja.msg".

qsmd.msg@@/main/0

qsmf.msg@@/main/0

qsmja.msg@@/main/0

[REDACTED] joint@@/main/6

[REDACTED] Add 8.1.4 RepAPIError.properties translation entries

Added file element "RepAPIErrorD.properties".

Added file element "RepAPIErrorF.properties".

Added file element "RepAPIErrorJA.properties".

RepAPIErrorD.properties@@/main/0

[REDACTED] RepAPIErrorF.properties@@/main/0

[REDACTED] RepAPIErrorJA.properties@@/main/0

=====

Short regress has 0 dif - run with use\_ism=false.

tk0irddl.suc	tkagrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrplsl.suc	tkogbrqr.suc	tkprdep1.suc	tkqrsels.suc
tkagqbd2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtrses.suc
tkagbqe1.suc	tkhodbck.suc	tkpasr81.suc	tkprdep3.suc	tkqrwhes.suc
tkagbqe2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkagqdb1.suc	tkhoucmr.suc	tkpbdl.suc	tkprdep5.suc	tkrcrash.suc
tkagqdb2.suc	tkigfsrt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkagqdqa.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrlmlite.suc
tkagqdrc.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkagqmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsqlp.suc	tkrsubs.suc
tkagqmbel.suc	tkoodcta.suc	tkpgcafs.suc	tkprttime.suc	tktrclus.suc
tkagqnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmnps.suc
tkagqnsb1.suc	tkoottcl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkagqnsb2.suc	tkoottci.suc	tkpgjols.suc	tkqrccats.suc	tktrtabs.suc
tkagqnsb3.suc	tkoottcm.suc	tkpgmain.suc	tkqrdatas.suc	tkrrexpss.suc
tkagqnsb4.suc	tkoottir.suc	tkpqords.suc	tkqrrexpss.suc	txxabrcb.suc
tkagqnsb5.suc	tkoodxu0.suc	tkpqgrgs.suc	tkqrgrps.suc	tkzrlso0.suc
tkagoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	
tkagrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	

Long regress has been started.

Transaction: rjenkins\_bug-702786

```

kkdc.c@@/main/85
[REDACTED] Merge From /vobs/rdbms/src/server/dict/dictlkup/kkdc.c@@/main/84
kkpoc.c@@/main/17
[REDACTED] more stuff
kkpam.h@@/main/13
[REDACTED] 702786: make kkpamKRange return fragment numbers
kdic.c@@/main/131
[REDACTED] Merge From /vobs/rdbms/src/server/ram/index/kdic.c@@/main/130
kkpam.c@@/main/19
[REDACTED] 702786: merge changes from andre
kkpam0.h@@/main/8
[REDACTED] 702786: merging in andre's changes
delexe.c@@/main/163
[REDACTED] try again
kxcc.h@@/main/15
[REDACTED] Merge From /vobs/rdbms/src/server/sqllang/if/kxcc.h@@/main/14
kxcc.c@@/main/23
[REDACTED] Merge From /vobs/rdbms/src/server/sqllang/integ/kxcc.c@@/main/22
xty.c@@/main/189
[REDACTED] Merge From /vobs/rdbms/src/server/sqllang/typeconv/xty.c@@/main/188

```

Transaction: thoang\_partobj\_bugs

```

atb.c@@/main/250
[REDACTED]
koke.h@@/main/28
[REDACTED] Add fragno argument to kokeicd2m()

```

- ktbl.c@@/main/38  
[REDACTED] ktblcls1: setup lsinfo for varray stored as lob  
koke.c@@/main/45  
[REDACTED]: Pass fragno to kokeicd2m()  
kaf.c@@/main/78  
[REDACTED] Pass fragno to kokeicd2m()  
updexe.c@@/main/231  
[REDACTED] Pass fragno to kokeicd2m

Out of Transaction changes:

.labellog@@/main/261  
[REDACTED] new log  
.manifest@@/main/solaris/268  
[REDACTED] new label  
.labellog@@/main/260  
[REDACTED] Correcting root directory entries for 980925 label.

=====

Y

X

V

X

500

2

This label was compiled with -xprofile=use option.

Short regress has 0 dif - run with use ism=false.

tk0irdd1.suc	tkaqrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriott.suc	tkb2srg.suc	tkommco2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrplsl.suc	tkogbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkaqpdq2.suc	tkgseces.suc	tkp83cl.suc	tkprdep2.suc	tkqrtrres.suc
tkaqpeq1.suc	tkhodbck.suc	tkpasr81.suc	tkprdep3.suc	tkqrwhes.suc
tkaqbeg2.suc	tkhommg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkaqgdq1.suc	tkhoucmr.suc	tkpbdl.suc	tkprdep5.suc	tkrcrash.suc
tkaqgdq2.suc	tkigfsrt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkaqgdpa.suc	tkirenb.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkaqgdrc.suc	tkirinds.suc	tkppro.suc	tkprmisc.suc	tkrmliti.suc
tkaqgmbq.suc	tkoodcat.suc	tkpotta.suc	tkprsqlp.suc	tkrsrsub.suc
tkaqgmei.suc	tkoodcta.suc	tkpqcafs.suc	tkprtime.suc	tktrcius.suc
tkaqgnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmpns.suc
tkaqnsb1.suc	tkoodctl.suc	tkpqgrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpqjois.suc	tkqrccats.suc	tktrtabs.suc
tkaqnsb3.suc	tkoodtem.suc	tkpqmain.suc	tkqrdat.suc	tktrvries.suc
tkaqnsb4.suc	tkoodtir.suc	tkpqord.suc	tkqrrexp.suc	txkxabrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqgrrs.suc	tkqrgrps.suc	txkxafini.suc
tkaqoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	txkxainit.suc
tkagrawi.suc	tkoolgvd.suc	tkprbugs.suc	tkqrnuls.suc	tkzr1soo.suc

Long regress will be started tomorrow evening.

Transaction: ramkrish bugfix-varray

kkbl.c@@/main/30  
[1/1] kkblalci: fix varray bug introduced as part of atbmov

Transaction: sputakki bug=511898 5

gerix.c@main/128 [REDACTED] Fix regression from merge of 511898

Transactions: sweng\_bug-702172 1

kda 600/main/113

Transaction: mkrishna fix obiview bugs

kxti.h@@/main/5  
[REDACTED] add KXTDF\_NTT\_TRIGGER  
kxti.c@@/main/14  
[REDACTED]  
kxto.c@@/main/22  
[REDACTED]: fix bug 704081  
kkdo.h@@/main/20  
[REDACTED] make lint happy  
kokv.c@@/main/20  
[REDACTED] fix core dumps when invalid type is given  
nsqcbc.c@@/main/53  
[REDACTED] nsomrge: set up froljc for dependent tables  
xtv.c@@/main/180  
[REDACTED]  
cvw.c@@/main/105  
[REDACTED] fix cvwosfor  
delexe.c@@/main/155  
[REDACTED] change kxtifrw calling

**Transaction: amozes\_lrg-25369**

dbsdrv.c@@/main/162  
[REDACTED] check if database is open before calling ksxshut

**Transaction: sdas\_bug-703264**

kau.c@@/main/90  
[REDACTED] partitioned iot: fix bug 703264  
kdu.c@@/main/115  
[REDACTED]: kdudcp(): replace kduusflag with kduusdfag  
kau.h@@/main/36  
[REDACTED] name/interface chg: kauibokey->kauibokey, kaubnikey->kauibnkey  
delexe.c@@/main/156  
[REDACTED] iot: set kduikauc in delini()  
updexe.c@@/main/221  
[REDACTED] iot: remove kdkreb call, kauibokey did its job  
qerfu.c@@/main/23  
[REDACTED] iot: set kduikauc in qerfuInitFrame()

**Transaction: bnnguyen\_bug-678044**

psdicd.c@@/main/174  
[REDACTED] bug678044

**Transaction: svedala\_create\_part\_demo**

demo@@/main/19  
[REDACTED] Added file element "cdemoplbc.c".  
Added file element "cdemoplbc.sql".  
Added file element "cdemoplbc.h".  
Added file element "cdemoplbc.dat".  
Removed file element "cdemoplbc.h".  
Removed file element "cdemoplbc.sql".  
Removed file element "cdemoplbc.dat".  
cdemoplbc.c@@/main/1

Transaction: amganesh\_ktprundo-noret  
ktpr.c@@/main/13  
[REDACTED]: ktprundo doesnt return

Transaction: thoang\_merge\_696471

kgl.h@@/main/205  
[REDACTED]: Removed kgldclp  
kg12.c@@/main/109  
[REDACTED]: Removed kgldclp  
kkm.c@@/main/380  
[REDACTED]:  
kkpod.c@@/main/27  
[REDACTED]  
atb.c@@/main/228  
[REDACTED]:  
cvw.c@@/main/106  
[REDACTED]  
dix.c@@/main/52  
qkadrv.c@@/main/209  
[REDACTED]

Transaction: nmacnaug\_merge\_9

kcbz.h@@/main/46  
[REDACTED]: do not use kcbhbba field directly  
kcl.c@@/main/113  
[REDACTED]: rename structure element to avoid conflict  
kcl0.h@@/main/25  
[REDACTED]: rename structure element to avoid conflict

Transaction: st\_plsql\_smkrishn\_no\_rebind

kkxwtp.c@@/main/192  
[REDACTED]: Merge From /vobs/rdbms/src/server/progint/opi/kkxwtp.c@@/main/188

Transaction: nmacnaug\_fix\_24

kcl.h@@/main/53  
[REDACTED]: add exclusive hint  
kcbz.c@@/main/119  
[REDACTED]: use exclusive hint  
kc12.h@@/main/34  
[REDACTED]: send class correctly to bsp  
kcl.c@@/main/111  
[REDACTED]: send class correctly to bsp

Transaction: lkaplan\_change\_drop

FDSAccess.java@@/main/5

[REDACTED]  
FDSObject.java@@/main/5  
[REDACTED]:

Transaction: nmacnaug\_lrg-25129

kcl2.h@@/main/35  
[REDACTED]: remove queued upconvert  
kcl.c@@/main/112  
[REDACTED]: remove queued upconvert  
kcl0.h@@/main/24  
[REDACTED]: remove queued upconvert

Transaction: jklein\_bug-708701

kdd.c@@/main/83  
[REDACTED]: bug 708701 - maintain col\_list between kdudnk calls.

Transaction: st\_plsql\_smkrishn\_backout\_rdbms

kkxwtp.c@@/main/194  
[REDACTED]: Temporarily back out no rebind

Transaction: mcusson\_logmn\_r\_nfy

krv.h@@/main/11  
[REDACTED]: Fix short regress problem caused by krvnfy()  
krvr.c@@/main/6  
[REDACTED]: Fix short regress problem caused by krvnfy()

Out of Transaction changes:

.depprod@@/main/solaris/184  
[REDACTED] update nlsrtl, network, spatial, slax, oracore, /vobs/oracle & precomp  
.labellog@@/main/212  
[REDACTED] new log  
.manifest@@/main/solaris/219  
[REDACTED]: new label  
dbfmig.c@@/main/osds/unix/solaris/1  
[REDACTED]:  
kkxwtp.c@@/main/193  
[REDACTED]: Back out psdscp, psdrpc until pfrrun.c is changed

---

Short regress has 0 dif - run with use\_ism=false.

tk0irddl.suc	tkaqrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrplsl.suc	tkoqbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkagbqd2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtrres.suc
tkagbqe1.suc	tkhodbck.suc	tkpavr81.suc	tkprdep3.suc	tkqrwhes.suc
tkagbqe2.suc	tkhomrg.suc	tkpasr0.suc	tkprdep4.suc	tkqxeisr.suc
tkagdq1.suc	tkhoucmr.suc	tkpbdl.suc	tkprdep5.suc	tkrcrash.suc
tkagdq2.suc	tkigfst.suc	tkpmsl1.suc	tkprdep6.suc	tkrcrsha.suc
tkagdqba.suc	tkirenbs.suc	tkpmsrl.suc	tkpreexec.suc	tkrmlite.suc
tkagdrc.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmlite.suc
tkagmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsglp.suc	tksrsub.suc
tkagmbei.suc	tkoodcta.suc	tkpcacfs.suc	tkprtime.suc	tktrclus.suc
tkagmnav.suc	tkoodecn.suc	tkpgdups.suc	tkprtimmm.suc	tktrmnps.suc
tkagnsbl.suc	tkoodecl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkagnsbs2.suc	tkoodecti.suc	tkpgjois.suc	tkqrcats.suc	tktrtabs.suc
tkagnsbs3.suc	tkoodecm.suc	tkpgmain.suc	tkqrdat.suc	tktrvies.suc
tkagnsbs4.suc	tkoodecir.suc	tkpgord.suc	tkqrexp.suc	tkxabrc.suc
tkagnsbs5.suc	tkoodoxu0.suc	tkpgrgrs.suc	tkqrgrps.suc	tkxafini.suc
tkagoc12.suc	tkoolqyc.suc	tkpgwhrs.suc	tkqrjoi.suc	tkxainit.suc
tkagrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	tkzrlso0.suc

Long regress will not be started.

#### Transaction: smuralid\_bugs2

kokq.c@@/main/14  
 [REDACTED]: kokqtpo, kokgbpo: propagate OPNOREF  
 koks.c@@/main/52  
 [REDACTED]: koksrbq: follow qbcnxt only if NOT(is\_first)

#### Transaction: liwong\_add\_error\_23473

e19400.msg@@/main/60  
 [REDACTED]: Add 23473

#### Transaction: dmwong\_approlemmsg

e24280.msg@@/main/72  
 [REDACTED]: add application role err msg

#### Transaction: bdagevil\_ppwj\_no\_hj\_buffout

qerhj.h@@/main/20  
 [REDACTED]: change flag BUFFER\_QKNJO to NO\_BUFFER\_QKNJO  
 qkna.h@@/main/14  
 [REDACTED]: change flag BUFFER\_QKNJO to NO\_BUFFER\_QKNJO  
 qerhj.c@@/main/46  
 [REDACTED]: buffer output if BUFFER\_QKNJO is set  
 qkadrv.c@@/main/205  
 [REDACTED]: qkadrv2(): add phase to set HJ output buff requirement  
 qkajoi.c@@/main/133

[REDACTED] : remove every reference to qkanbf()

Transaction: masubram\_masubram\_null\_refl\_1

c0800050.sql@@/main/3  
[REDACTED]  
d0800050.sql@@/main/30  
[REDACTED]  
sql.bsq@@/main/202  
[REDACTED]  
kkdl.c@@/main/360  
[REDACTED] : Merge From /vobs/rdbms/src/server/dict/dictlkup/kkdl.c@@/main/356  
kkz.h@@/main/46  
[REDACTED]  
kkzd.h@@/main/12  
[REDACTED]  
kkzf.h@@/main/12  
[REDACTED]  
kkzi.h@@/main/5  
[REDACTED]  
kkzu.h@@/main/10  
[REDACTED]  
kkzv.h@@/main/5  
[REDACTED]  
kkzd.c@@/main/19  
[REDACTED]  
kkzf.c@@/main/26  
[REDACTED]  
kkzi.c@@/main/12  
[REDACTED]  
kkzu.c@@/main/12  
[REDACTED]  
kkzv.c@@/main/15  
[REDACTED]  
prvtsnap.sql@@/main/27  
[REDACTED]  
knt.c@@/main/13  
[REDACTED] : Merge From /vobs/rdbms/src/server/repl/trigger/knt.c@@/main/12  
qsmqutl.c@@/main/5  
[REDACTED]  
kkz.c@@/main/126  
[REDACTED]

Transaction: bgoyal\_bug-692581

e29250.msg@@/main/92  
[REDACTED] add 30555

Transaction: qyu\_bug-428835

[REDACTED]  
catalog.sql@@/main/302  
[REDACTED]

Transaction: vkarra\_tpcd\_changes

kdi.h@@/main/73

[REDACTED]: Merge From /vobs/rdbms/src/server/ram/if/kdi.h@@/main/72  
kdis.h@@/main/17  
[REDACTED]: add variables for leaf and branch splits  
kdi.c@@/main/168  
[REDACTED]  
kdis.c@@/main/52  
[REDACTED]: Merge From /vobs/rdbms/src/server/ram/index/kdis.c@@/main/51

**Transaction: akruglik\_bug-696737**

e12700.msg@@/main/53  
[REDACTED]: add message 14176  
ktbl.c@@/main/29  
[REDACTED]: fix for bug 696737: if performing PITR, allow STORAGE clause when parsing def  
prsc.c@@/main/387  
[REDACTED]: fix for bug 696737: do not restrict attributes which may be specified for a H

**Transaction: sichandr\_optimize\_kokbint**

kokb.c@@/main/16  
[REDACTED]  
kprb.h@@/main/14  
[REDACTED]: add kprbuu\* (callback bind) support for ADTs

**Transaction: aksrivas\_bug-691448**

kge.h@@/main/41  
[REDACTED]: fix bug 691448, add KGENMFASSERT macro and kgearnmfe proto  
kge.c@@/main/59  
[REDACTED]: fix bug 691448, add named fatal assert routine  
kjga.h@@/main/7  
[REDACTED]: fix bug 691448, add named fatal assert macros

**Transaction: hasun\_handle\_grpbys\_cols\_correctly\_for\_mavs**

kkz.c@@/main/127  
[REDACTED]: Use a unique index for MAVs instead of PK constraint  
kkzg.c@@/main/28  
[REDACTED]: Modify kkzgindcb() to generate unique indexes for MAVs

**Transaction: ramkrish\_ramkrish\_iotlob\_atbmov**

kkm.c@@/main/379  
[REDACTED]  
kkpo.c@@/main/40  
[REDACTED]  
atb.c@@/main/227  
[REDACTED]  
kkb.c@@/main/82  
[REDACTED]  
kokl2.c@@/main/29  
[REDACTED]  
kditc.h@@/main/10  
[REDACTED]  
kdic.c@@/main/113

[REDACTED]

[REDACTED]  
send1.c@@/main/73  
[REDACTED]  
prsdef.h@@/main/35  
[REDACTED]  
ctc.c@@/main/221  
[REDACTED]  
kkbl.c@@/main/28  
[REDACTED]  
prsc.c@@/main/386  
[REDACTED]:

Transaction: **mkamath\_errmsg**  
e24280.msg@@/main/73  
[REDACTED] Adding error message 25261

Transaction: **sbedarka\_bug-664195\_1**  
catalog.sql@@/main/301  
[REDACTED]

Transaction: **sbedarka\_bug-683833\_1**  
kkdc.c@@/main/81  
[REDACTED]

Transaction: **sbedarka\_bug-475585\_1**  
kvpf.c@@/main/11  
[REDACTED]

Transaction: **lkaplan\_sndcmp\_delete**  
columnDescriptor.java@@/main/6  
[REDACTED]

Transaction: **clei\_bug-696853\_1**  
kzra.c@@/main/8  
[REDACTED]

Transaction: **lbarton\_bug-654891**  
exuevw.c@@/main/25  
[REDACTED] Use explicit array index in offsetof

Transaction: **dalpern\_snapshot\_purity\_adjustments\_errmsg**  
e29250.msg@@/main/91  
[REDACTED] base purity decisions on DETERMINISTIC

[REDACTED]

**Transaction: gtaracha\_bug-566533**

opndef.h@@/main/124  
[REDACTED] Removing OPTHTOR and OPTATOH  
odfdef.c@@/main/82  
[REDACTED] 3: Fixing bug 566533  
koke.h@@/main/25  
[REDACTED] (566533): Removing kokeehtx, kokeihtx  
koke.c@@/main/39  
[REDACTED] (566533): Removing kokeehtx, kokeihtx and OPTATOH from kokeithx  
kkzp.c@@/main/24  
[REDACTED] Removing OPTHTOR and OPTATOH

**Transaction: nramani\_ldap\_errors**

e24280.msg@@/main/74  
[REDACTED] adding ldap integration errors

Out of Transaction changes:

.labellog@@/main/209  
[REDACTED] new log  
.manifest@@/main/solaris/216  
[REDACTED] new label  
ctc.c@@/main/222  
[REDACTED] fix comp err

---

Short regress has 0 dif - run with use\_ism=false.

tk0irddl.suc	tkaqrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrpls1.suc	tkogbrqr.suc	tkprdepl.suc	tkqrsebs.suc
tkagbdq2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtrres.suc
tkagbq1.suc	tkhodbck.suc	tkpasr81.suc	tkprdep3.suc	tkqrwbes.suc
tkagbq2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkagdq1.suc	tkhoucmr.suc	tkpbdl.suc	tkprdep5.suc	tkrcrash.suc
tkagdq2.suc	tkigfart.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkagdqba.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkagdrc.suc	tkrinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkaqmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsqlp.suc	tkrsrsub.suc
tkaqmbei.suc	tkoodcta.suc	tkpqcafs.suc	tkprtime.suc	tktrclus.suc
tkaqmnav.suc	tkoodctn.suc	tkpqdups.suc	tkprtimm.suc	tktrmmps.suc
tkaqnsbl.suc	tkooldt1.suc	tkpqgrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkooldtci.suc	tkpqjois.suc	tkqrccats.suc	tktrtabs.suc
tkaqnsb3.suc	tkooldtcm.suc	tkpqmain.suc	tkqrdat.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpqgords.suc	tkqrrexp.suc	tkxabbrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqgrgs.suc	tkqrgrps.suc	tkzrls00.suc
tkaqoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	
tkaqrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	

Long regress has been started.

Transaction: **svedala\_fix\_cdemosyev**

cdemosyev.c@@/main/st\_rdbms\_8.1.4/1

Transaction: **gbhatia\_define\_net\_use\_ldap\_flag**

s\_rdbms.mk@@/main/osds/unix/solaris/st\_rdbms\_8.1.4/1  
Define NET\_USE\_LDAP flag

Transaction: **mtakashi\_merge\_jox\_981022**

jox.c@@/main/st\_rdbms\_8.1.4/8  
merge from cvs

Transaction: **skmishra\_fix-dif-4**

opiodr.c@@/main/st\_rdbms\_8.1.4/4

Out of Transaction changes:  
.depprod@@/main/solaris/st\_rdbms\_8.1.4/15  
update javavm  
.labellog@@/main/st\_rdbms\_8.1.4/15

[REDACTED] Oct 00 new log  
.manifest@@/main/solaris/st\_rdbms\_8.1.4/16  
[REDACTED]: new label  
rdbmsqaenv@@/main/st\_rdbms\_8.1.4/2  
[REDACTED]: Added entries to T\_SOURCE, T\_COM, etc. to stay in sync with .ndeprodenv  
=====

[REDACTED]

Short regress has 0 dif - run with use\_ism=false.

tk0irddl.suc	tkb2srg.suc	tkogbrqr.suc	tkprdemo.suc	tkgrsels.suc
tk0iriot.suc	tkdrplsl.suc	tkp83cl.suc	tkprdep1.suc	tkqrtrs.suc
tk0irmts.suc	tkgrsecs.suc	tkpasc81.suc	tkprdep2.suc	tkgrwhes.suc
tkaqbdq2.suc	tkhodbck.suc	tkpash81.suc	tkprdep3.suc	tkgxieisr.suc
tkaqbqeq1.suc	tkhosrg.suc	tkpasrg0.suc	tkprdep4.suc	tkrcrash.suc
tkaqbqeq2.suc	tkhoucsr.suc	tkpbdl.suc	tkprdep5.suc	tkrcrsha.suc
tkaqdqb1.suc	tkigfrst.suc	tkpdlipo.suc	tkprdep6.suc	tkrmlite.suc
tkaqdqb2.suc	tkirenbs.suc	tkpmsrl1.suc	tkprexec.suc	tkrmliti.suc
tkaqdqb4.suc	tkirinds.suc	tkpmsrnl.suc	tkprmisc.suc	tkrsrubs.suc
tkaqdqrc.suc	tkoodcat.suc	tkpopro.suc	tkprsqlp.suc	tktrclus.suc
tkaqmhdq.suc	tkoodcta.suc	tkpotta.suc	tkprttime.suc	tktrrnmps.suc
tkaqmbel.suc	tkoodctn.suc	tkpqcafs.suc	tkprtimm.suc	tktrnaos.suc
tkaqmnav.suc	tkoodtcl.suc	tkpqdups.suc	tkpuori.suc	tktrtabs.suc
tkaqnsb1.suc	tkoodtci.suc	tkpqgrps.suc	tkqrccats.suc	tktrvies.suc
tkaqnsb2.suc	tkoodtcm.suc	tkpqjois.suc	tkqrdat.suc	tkxabrch.suc
tkaqnsb3.suc	tkoodtir.suc	tkpqmain.suc	tkqrrexp.suc	tkzrls0.suc
tkaqnsb4.suc	tkoodxu0.suc	tkpqord.suc	tkqrgrps.suc	tkzrlso0.suc
tkaqnsb5.suc	tkoolqyc.suc	tkpqgrs.suc	tkqrjois.suc	
tkaqoc12.suc	tkoolqyd.suc	tkpqwhrs.suc	tkqrnuls.suc	
tkaqrawi.suc	tkoolqyi.suc	tkprbugs.suc	tkqrord.suc	
tkaqrawt.suc	tkoommo2.suc	tkprddls.suc	tkqrsecb.suc	

Long regress has been started.

Transaction: nvishnub\_bug-786151

exuept.c@@/main/14

Transaction: gtarora\_kollasg\_OCIOObjectSetAttr

orid.c@@/main/8  
lint

Transaction: anithrak\_fix\_state\_obj\_dump

kcb.h@@/main/167  
[REDACTED] Add new where for kcbmbl for call from kcbzps  
kcbz.c@@/main/149

Transaction: whe\_fix\_lrg31076

or1.c@@/main/18

[REDACTED]: lrg31076:typo in OCIOPaqueCtxGetHandles

**Transaction: jfeenan\_merger**

qsmqutl.c@@/main/13

[REDACTED]: reduce invalidation and lock window

**Transaction: fge\_bug-705269**

qkadrv.c@@/main/256

[REDACTED] #(705269): qkadrv projects rwo for bitmap dn if ROWNUM

qkna.c@@/main/25

[REDACTED] #(705269): qknpxAllocate: new rwo if bitmap dn underneath

**Transaction: gbhatia\_ano\_radius\_encrypt\_passwd**

kpuzln.c@@/main/31

[REDACTED]: ANO Radius Fix

upilog.c@@/main/63

[REDACTED]: ANO Radius Fix

tzia.c@@/main/14

[REDACTED]: ANO Radius Fix

Out of Transaction changes:

.depprod@@/main/solaris/293

[REDACTED] update plsql, precomp, sqlplus, ordts, OEMAGENT

.labellog@@/main/334

[REDACTED] new log

.manifest@@/main/solaris/341

[REDACTED] new label

---